1

Introduction to DotNetNuke and Resources

COPYRIGHTED MATERIAL

Anyone can visit `DotNetNuke.com` and download a copy of the DotNetNuke development framework, but it is a very daunting task to fully understand the different versions, packages, and resources available. This chapter provides an overview of DotNetNuke versions, packages, and resources — those available via `DotNetNuke.com` and other community sites.

This chapter begins with an examination of the three major DotNetNuke versions available, and details a bit of the history surrounding how DotNetNuke has evolved to the framework that it is today. Technology and other requirements are discussed for key releases in recent DotNetNuke history. The discussion then examines the various download packages available from `DotNetNuke.com`, and the process for obtaining legacy versions, should the need arise for a noncurrent version. The chapter concludes with a quick overview of community resources available for more information on DotNetNuke and module development with DotNetNuke.

Versions Explained

A first-time visitor to the DotNetNuke website can easily be confused by all the versions of DotNetNuke. As of this writing, the `DotNetNuke.com` website has published links for downloading versions 4.8.4, 4.70, 4.62, and 3.3.7. This alone can be very confusing to many, as identifying the proper starting point can be a bit unclear. The following sections examine the 3.x, 4.x, and 5.x versions of DotNetNuke, with key milestones identified and details on when and why the downloads are available.

The DotNetNuke Corporation provides *all* historical versions for download, but they can be a bit complicated to locate. All historical version downloads can be found via the "Legacy Releases" section of the Downloads page, which can direct you to the SourceForge listings for all DotNetNuke downloads.

3.x Versions

The 3.x DotNetNuke platform is a legacy platform that is designed to work with ASP.NET 1.1. The final release of the 3.x platform was version 3.3.7, released November 30, 2006. The new use of this platform is not recommended because many security and performance enhancements have been completed in subsequent releases.

Users currently on the DotNetNuke 3.x platform can follow standard upgrade processes to upgrade installations to the most current DotNetNuke releases. In many cases, this is recommended via a multi-step upgrade process. The exact process for upgrading is beyond the scope of this book, but Appendix A provides community resources with upgrade guides.

4.x Versions

The 4.x DotNetNuke platform was the first version to be built for the .NET 2.0 framework. Releases prior to 4.3.7 were done in parallel to the 3.x platform to allow the option of running .NET 1.1 or 2.0. Starting with version 4.4.0, the 4.x version was the only supported version. During the 4.x life cycle, DotNetNuke issued many releases with greatly improved core functionality.

Following are some highlighted versions:

- **Version 4.4.0:** Released in December 2006, Version 4.4.0 provided major performance and scalability improvements. After upgrading, many users noticed dramatic performance improvements. With this being the first .NET 2.0–only release, many 3.x users were encouraged to upgrade to this version to attain the performance benefits.
- **Version 4.5.0:** Released in April 2007, version 4.5.0 provided many usability and performance enhancements to the framework. The most notable enhancement with this release was the inclusion of ASP.NET Ajax integration into the core framework. This integration enables module developers to implement Ajax functionality with actions as simple as checking a box to have controls automatically wrapped in an update panel.
- **Version 4.6.0:** Released in September 2007, version 4.6.0 was announced as a stabilization release. However, it included a few somewhat hidden enhancements that enabled developers to produce more efficient modules. Most notable was the inclusion of the IHydratable interface for object hydration. (IHydratable is discussed in detail in Chapter 10.) Another key enhancement with this release was the default inclusion of the FCK editor that replaced the older FreeTextBox editor for all rich text editing inside a DotNetNuke portal.
- **Version 4.6.2:** Released September 2007, version 4.6.2 is worth noting because it contains a mostly unknown feature, `XMLMerge`, which enables individuals to upgrade to any future version of DotNetNuke without manually merging `web.config` files. This is a major administrative benefit because the risk for failed upgrades is dramatically reduced when manual `web.config` changes are no longer required.
- **Version 4.8.0:** Released December 2007, version 4.8.0 provided support for Visual Studio 2008 and Internet Information Server (IIS) 7.0 compatibility for the entire framework. The versions released between 4.5.0 and 4.8.0 were stabilization releases that helped to solidify the DotNetNuke core, and created the solid base needed to create DotNetNuke 5.0.

- **Security Releases:** As of this writing, a number of security point releases (for example, 4.8.3 and 4.8.4) where completed to resolve security issues identified in prior versions of the framework. Because of the security issues resolved in these versions, DotNetNuke Corporation strongly recommended that users upgrade to the most current point version to resolve any potential security concerns. For detailed information regarding security notices and other security concerns, visit the DotNetNuke Security Policy page at `www.dotnetnuke.com/News/SecurityPolicy/tabid/940/Default.aspx`.
- **Version 4.9.0:** Released September 2008, version 4.9.0 provided a few additional security fixes, as well as a few back-ported features from the upcoming 5.0 release. Special items to note in this release include a new default skin and an updated version of the FCK editor control providing support for the Safari browser.

5.x Versions

The DotNetNuke 5.x platform includes many major enhancements to the core DotNetNuke system. This includes a unified package system for modules, skin objects, skins, and more, as well as separation of core administration modules to provide users with greater flexibility in site management than was possible in prior versions.

> **As of this writing, the exact date of the first 5.x release is unknown, but it is likely to be available by the time you read this. All topics covered in this book relate to the 5.x version. Differences between the 5.x version and previous versions are pointed out where appropriate.**

Packages Explained

Each version of DotNetNuke contains four different packages that may be downloaded for use. Knowing the differences between the packages and which one to use is key to successfully working with and developing for DotNetNuke. The following sections explain the differences between the packages, as well as the factors governing when each should be used.

Starter Kit

The Starter Kit is a Visual Studio Installer Package (`.vsi`) used to add DotNetNuke-specific templates to your Visual Studio installation. You can find more detailed information on Visual Studio Installer packages at the Microsoft Developers Network (MSDN) at `http://msdn.microsoft.com/en-us/library/ms185314(VS.80).aspx`.

The following template types are included with the Starter Kit:

> **The following descriptions discuss multiple project types and are provided for reference only. Chapter 2 provides a deeper look at the differences between project types.**

- **DotNetNuke Web Application Framework:** This template is the entire DotNetNuke solution that can be used to set up a website for development. This template should not be used because development can more easily be completed via manual installations using the Install package.
- **DotNetNuke Compiled Module (VB):** This template will create a Web Application Project (WAP)-compiled Visual Basic (VB) module. This template creates all standard DotNetNuke module elements, including the data access layer (DAL) and multiple view controls. This is the template that is used in this book for all VB examples. It creates a module that will be compiled into a single `.dll` for distribution via standard build processes.
- **DotNetNuke Dynamic Module (VB):** This template will create a Web Site Project (WSP) VB module that utilizes the `app_code` folder for all classes. This template creates a complete module with similar contents to the Compiled Module template, including full DAL classes. This template is recommended for users with Visual Studio Express, which prior to Visual Studio 2008 Service Pack 1 cannot support the WAP development model. Chapter 2 provides full information on Visual Studio versions and project types.

  This template is implemented as an item template, and may only be added to a DotNetNuke Web Application Framework solution created via the template discussed previously.

- **DotNetNuke Dynamic Module (C#):** This template will create a Web Site Project (WSP) C# module that utilizes the `app_code` folder for all classes. The created project will be exactly like that of the Dynamic VB Module, but translated to the C# language. This project template is recommended for developers who prefer C# and are using Visual Studio Express.

  This template is implemented as an item template and may only be added to a DotNetNuke Web Application Framework solution created via the template discussed previously.

- **DotNetNuke Simple Dynamic Module (VB):** This template will create a dynamic VB.NET module, but the module created will only have a single view control and the `.dnn` manifest. This is designed for individuals looking for a custom implementation, or those looking for a clean project to start with.

  This template is implemented as an item template and may only be added to a DotNetNuke Web Application Framework solution created via the template discussed previously.

- **DotNetNuke Skin:** The final template included in the Starter Kit creates all files needed to create a single skin and container for a DotNetNuke website. Chapter 3 discusses skins and containers in more detail.

  This template is implemented as an item template and may only be added to a DotNetNuke Web Application Framework solution created via the template discussed previously.

Source Package

The source package of DotNetNuke contains all source code for the DotNetNuke framework in a fashion that can be modified right out of the box. The downloaded ZIP file contains two folders: `Web site` and `library`. The `Web site` portion is the same as that provided with the Install version. The `library` section provides all additional source code used to create the compiled `DotNetNuke.*.dll` files that are included in installation and upgrade versions.

Recommended Uses

The use of this package of DotNetNuke is recommended only for individuals who are interested in customizing the DotNetNuke core. Note here that modifications to the DotNetNuke core will remove an installation from the upgrade path, and all changes must be managed on a case-by-case basis during upgrade.

> **In my experience, I have found that it is much easier to find other ways around a core modification that enable my sites to stay on the standard upgrade path.**

Benefits

Regardless of customizations to the core framework, this version will provide you with *all* source code for the DotNetNuke framework. Many organizations view this as a great benefit, as you have all source code available for review and/or modification, should business needs arise that require the code to be available.

Install Package

The Install package of DotNetNuke is exactly that: the minimal collection of files needed to run the DotNetNuke framework, including all core DotNetNuke modules (not all are installed by default), and compiled library assemblies. This package is used in Chapter 2 to create the development environment used throughout the remainder of the book.

Recommended Uses

This package of DotNetNuke is recommended for all production installations and all development environments that will *not* require modifications to the core DotNetNuke framework. Some individuals additionally use this version for upgrade purposes to ensure that core modules are upgraded at the same time as the core.

Benefits

Using this package has a number of benefits, but one of the best is that it is simply a less cluttered download. Because you only get the core files needed to install and use DotNetNuke, you are not bothered by other files that you will not modify.

Upgrade Package

The final DotNetNuke package available is the Upgrade package. This package is optimized to include *only* the files needed to upgrade to the specific version. This package does not contain a few key files that would cause problems with upgrades. The most notable difference is that this package does *not* contain a `web.config` file, although you do have a `release.config` file for reference, if needed, which is essentially a copy of the `web.config` file. It is named differently to avoid the accidental overwriting of an existing configuration.

Recommended Uses

The only recommended use for this package is to perform an upgrade to an existing DotNetNuke site. Note that depending on your existing DotNetNuke version, the upgrade process might not be as simple as overwriting files with the Upgrade package. See the available DotNetNuke documentation and community resources for complete information regarding DotNetNuke upgrades.

Benefits

The major benefit of this package is that it is a simple package that can be used to quickly update the core DotNetNuke framework, and associated items that need to be upgraded — namely, the Text/HTML module because it is part of the core. You will not be bothered with the extra overhead of updated core modules and install-only files. Along these same lines, this package will prevent many common errors that often occur when individuals upgrade using the Install package.

Resources

A project as large of DotNetNuke has many resources available — those provided by DotNetNuke Corporation and those provided by community members. Abundant though this information is for many people starting to develop in DotNetNuke, you may find yourself in a vexing cat-and-mouse game looking for the information specific to your needs. The following sections introduce many DotNetNuke and community resources that contain valuable tutorials and/or documentation.

DotNetNuke.com

The `DotNetNuke.com` website, shown in Figure 1-1, is the most commonly overlooked community resource. There is so much information on this site that many people give up before finding exactly what they need. The material presented here can help you zero in on the information you are looking for, and perhaps other items of interest.

Figure 1-1

From the home page, you can see that the navigational structure is clearly laid out for quickly getting to key items.

Downloads

Obviously, the Downloads link shown at the top of Figure 1-1 is one of the most active pages of the DotNetNuke website. Using this link, you can access downloads for all current and past versions of the DotNetNuke core and core modules. You must be registered and logged into the `DotNetNuke.com` website in order to view the downloads. Once you are logged in, you will notice that the downloads are divided into multiple sections:

- **Core Framework:** You can use the Core Framework download section to locate the different versions and packages referenced earlier in this chapter. Notes regarding the specifics of each version and package are also provided to ensure that the most appropriate package is downloaded.
- **Supplementary Downloads:** These components are downloads that are part of the core, but not part of the DotNetNuke Projects program. Items downloaded here are developed for use in the core framework, but do not have any other place for download.
- **DotNetNuke Projects (New Releases):** Downloads in this section include the most current releases of various core DotNetNuke module projects. The key thing to note about this section is that not all module downloads are found here; only the most current releases (for modules that have had recent releases) are found here. For modules not listed here, the module's project page is the place to go for downloads.
- **Legacy Releases:** The Legacy Releases section contains a single link to the `SourceForge.net` repository for all DotNetNuke downloads. Clicking this link brings up the window shown in Figure 1-2. Following links from this page, you can drill down to find any legacy release of any DotNetNuke project. This function is a very commonly overlooked aspect of the DotNetNuke downloads section.

Latest File Releases

Package	Release	Date	Notes / Monitor	Downloads
DotNetNuke®	04.08.04	June 11, 2008		Download
DotNetNuke® Announcements	04.00.01	January 27, 2008		Download
DotNetNuke® Auth: AD	01.00.03	March 19, 2008		Download
DotNetNuke® Auth: Cardspace	01.00.00	February 7, 2008		Download
DotNetNuke® Auth: LiveID	01.00.00	October 17, 2007		Download
DotNetNuke® Auth: OpenID	01.00.00	October 17, 2007		Download
DotNetNuke® Blog	03.04.00	January 8, 2008		Download
DotNetNuke® Chat	01.00.01	January 27, 2008		Download
DotNetNuke® Contacts	03.01.00	June 9, 2005		Download
DotNetNuke® Documents	04.00.00	April 4, 2008		Download

Figure 1-2

Forums

The DotNetNuke forums are a great technical resource. Many community members and core team members monitor the forums on a very regular basis to answer various questions and share information about numerous aspects of DotNetNuke. Typically, if you are encountering an issue when you are starting out with module development or any other task within the DotNetNuke framework, you can get a response within a day. The amount of information shared via this communication channel is quite amazing, and many new users can benefit greatly simply by reviewing past threads to learn the various tips and tricks shared.

Blogs

Blogs on `DotNetNuke.com` are written by core team members and project team leads. Typically, this communication channel is where you can get an idea of when the next module release is going to be, or what new features are going to be included in the next core release. Many helpful tutorials are posted, such as those from Michael Washington, Charles Nurse, and many others, which help expose the development community to sometimes little-known benefits of the DotNetNuke core.

Documentation

Documentation is available via the `DotNetNuke.com` website. However, as of this writing, the information contained in this section is a bit out of date, especially from a development standpoint. Nonetheless, much of it can be very helpful for those looking to understand how the core framework is put together. A few key documents include "Module Localization Guide," "Security: Hardening Installations," and "Security: Module Development." These documents address three key DotNetNuke areas of concern, and provide a great deal of information in a compact and easy-to-read format.

Online Help

Another great resource, available at `www.dotnetnuke.com/Resources/Knowledge/DotNetNukeOnlineHelp/tabid/787/Default.aspx`, is the online help documentation for the most current DotNetNuke version. This documentation is directed more toward site administrators and content management, but many advanced topics are covered to help answer questions regarding core DotNetNuke functionality. The DotNetNuke Corporation also offers a PDF version of this documentation for those looking for an offline solution.

Sponsorship Program

The DotNetNuke sponsorship program is one final resource that is worth mentioning. This sponsorship program enables individuals to attain community recognition for supporting the DotNetNuke project, as well as attain more direct access to DotNetNuke core team members for assistance via a private discussion forum. For individuals who want a more direct communication channel for questions and answers, while supporting the open-source initiative, this is the way to participate with mutual benefits.

Community Resources

There are many community resources that provide more focused DotNetNuke tutorials, videos, training, and other general help. Some resources are available free, while others have membership costs associated with them.

Following are a few resources that are key for an individual starting with development in DotNetNuke:

- `dnncreative.com`: This website is a subscription service that provides video tutorials related to site configuration, administration, troubleshooting, and development. Many individuals find the information contained on this site to be an invaluable resource. Membership to DNN Creative grants a user access to more than 300 historical tutorials, reviews, and more, as well as access to all upcoming content for the next year.
- `dotnetnuke.adefwebserver.com`: This website is maintained by Michael Washington, a long-standing DotNetNuke core team member. On this site you will find guided module development tutorials, each focusing on one key element. These quick-to-read tutorials are great for refreshers, or to quickly learn how to complete a specific task.
- `mitchelsellers.com`: This is my website, where I maintain a mostly DotNetNuke-focused blog. Any updates to this book after publication will also be included here. You will find detailed installation, upgrade, migration, backup, and development tutorials mixed in with relevant blog postings.

Appendix A contains a more comprehensive listing of available community resources.

Summary

This chapter has introduced the different versions and packages of DotNetNuke, as well as a few community resources that might help greatly along the way. Understanding the differences between the DotNetNuke versions is key to understanding dependencies and capabilities, which is a key component when distributing custom modules.

Now that you have a firm base knowledge of the DotNetNuke package options, you are ready to progress to the installation and configuration of the development environment, as well as a discussion about the available project types, all of which is covered in detail in Chapter 2.